Difference between Edge Computing and Fog Computing

In the world of data centers with wings and wheels, there is an opportunity to lay some work off from the centralized cloud computing by taking less compute intensive tasks to other components of the architecture. In this blog, we will explore the upcoming frontier of the web — **Edge Computing**.

What is the "Edge"?

The 'Edge' refers to having computing infrastructure closer to the source of data. It is the distributed framework where data is processed as close to the originating data source possible. This infrastructure requires effective use of resources that may not be continuously connected to a network such as laptops, smartphones, tablets, and sensors. Edge Computing covers a wide range of technologies including wireless sensor networks, cooperative distributed peer-to-peer ad-hoc networking and processing, also classifiable as local cloud/fog computing, mobile edge computing, distributed data storage and retrieval, autonomic self-healing networks, remote cloud services, augmented reality, and more.

Cloud Computing is expected to go through a phase of decentralization. Edge Computing is coming up with an ideology of bringing compute, storage and networking closer to the consumer.

But Why?

Legit question! Why do we even need Edge Computing? What are the advantages of having this new infrastructure?

Imagine a case of a self-driving car where the car is sending a live stream continuously to the central servers. Now, the car has to take a crucial decision. The consequences can be disastrous if the car waits for the central servers to process the data and respond back to it. Although algorithms like YOLO_v2 have sped up the process of object detection the latency is at that part of the system when the car has to send terabytes to the central server and then receive the response and then act! Hence, we need the basic processing like when to stop or decelerate, to be done in the car itself.

The goal of Edge Computing is to minimize the latency by bringing the public cloud capabilities to the edge. This can be achieved in two forms — custom software stack emulating the cloud services running on existing hardware, and the public cloud seamlessly extended to multiple point-of-presence (PoP) locations.

Following are some promising reasons to use Edge Computing:

1. **Privacy**: Avoid sending all raw data to be stored and processed on cloud servers.
2. **Real-time responsiveness**: Sometimes the reaction time can be a critical factor.
3. **Reliability**: The system is capable to work even when disconnected to cloud servers. Removes a single point of failure.

To understand the points mentioned above, let's take the example of a device which responds to a hot keyword. Example, Jarvis from Iron Man. Imagine if your personal Jarvis sends all of your private conversations to a remote server for analysis. Instead, It is intelligent enough to respond when it is called. At the same time, it is real-time and reliable.

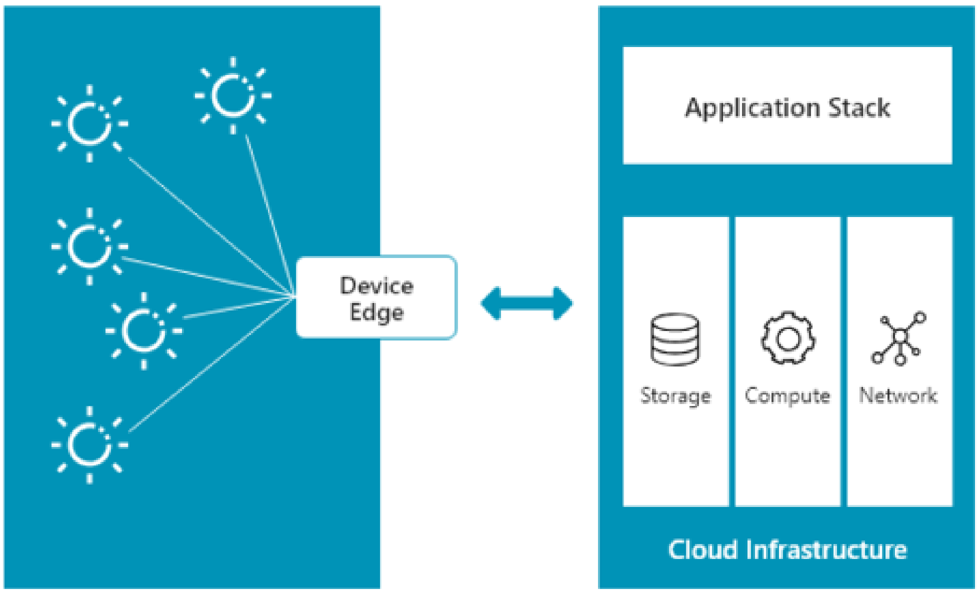
Intel CEO Brian Krzanich said in an event that autonomous cars will generate 40 terabytes of data for every eight hours of driving. Now with that flood of data, the time of transmission will go substantially up. In cases of self-driving cars, real-time or quick decisions are an essential need. Here edge computing infrastructure will come to rescue. These self-driving cars need to take decisions is split of a second whether to stop or not else consequences can be disastrous.

Another example can be drones or quadcopters, let's say we are using them to identify people or deliver relief packages then the machines should be intelligent enough to take basic decisions like changing the path to avoid obstacles locally.

Forms of Edge Computing

Device Edge

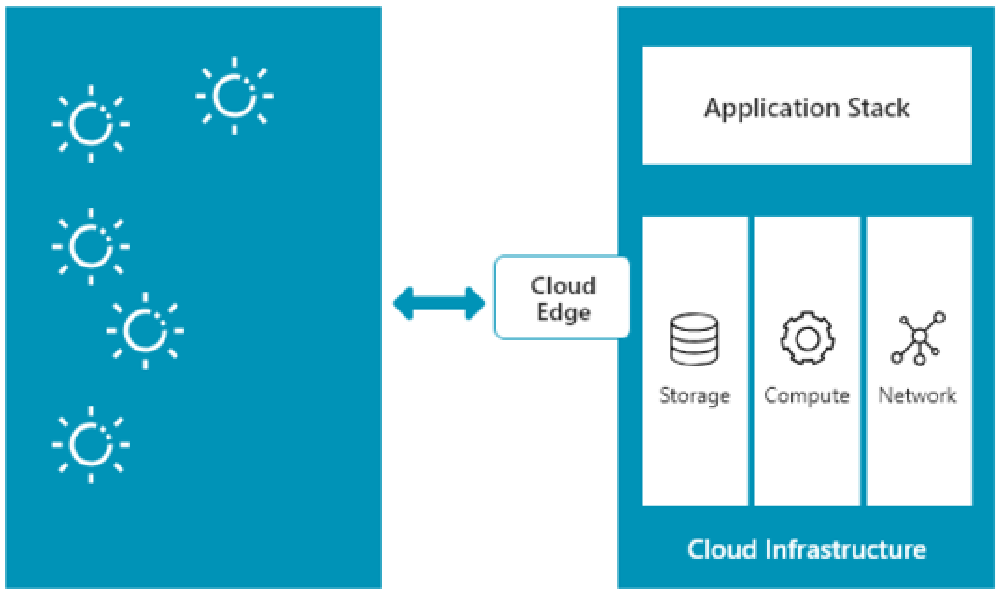
In this model, Edge Computing is taken to the customers in the existing environments. For example, AWS Greengrass and Microsoft Azure IoT Edge.



Cloud Edge

This model of Edge Computing is basically an extension of the public cloud. Content Delivery Networks are classic examples of this topology in which the static content is cached and delivered through a geographically spread edge locations.

Vapor IO is an emerging player in this category. They are attempting to build infrastructure for cloud edge. Vapor IO has various products like Vapor Chamber. These are self-monitored. They have sensors embedded in them using which they are continuously monitored and evaluated by Vapor Software, VEC(Vapor Edge Controller). They also have built OpenDCRE, which we will see later in this blog.



The fundamental difference between device edge and cloud edge lies in the deployment and pricing models. The deployment of these models — device edge and cloud edge — are specific to different use cases. Sometimes, it may be an advantage to deploy both the models.

Edges around you

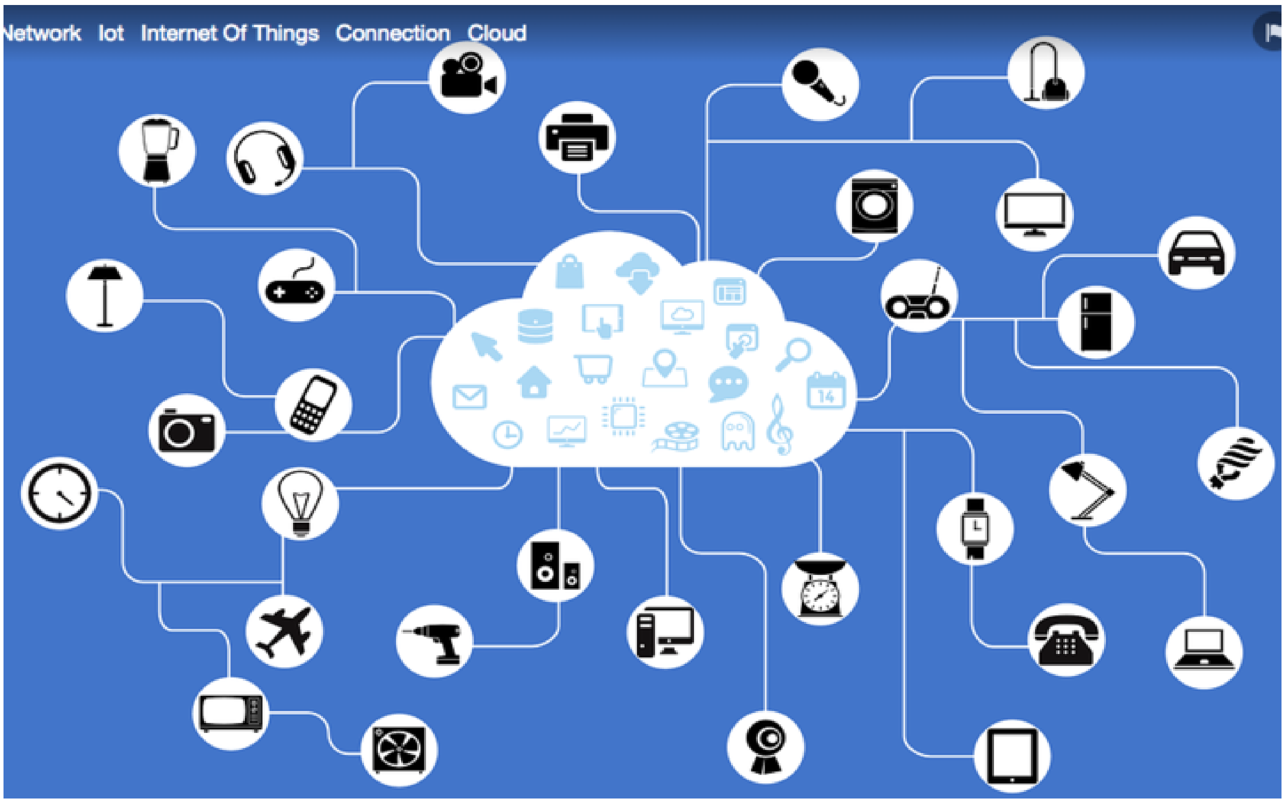
Edge Computing examples can be increasingly found around us:

1. Smart street lights
2. Automated Industrial Machines
3. Mobile devices
4. Smart Homes
5. Automated Vehicles (cars, drones etc)

Data Transmission is expensive. By bringing compute closer to the origin of data, latency is reduced as well as end users have better experience. Some of the evolving use cases of Edge Computing are Augmented Reality(AR) or Virtual Reality(VR) and the Internet of things. For example, the rush which people got while playing an Augmented Reality based pokemon game, wouldn't have been possible if "real-timeliness" was not present in the game. It was made possible because the smartphone itself was doing AR not the central servers. Even Machine Learning(ML) can benefit greatly from Edge Computing. All the heavy-duty training of ML algorithms can be done on the cloud and the trained model can be deployed on the edge for near real-time or even real-time predictions. We can see that in today's data-driven world edge computing is becoming a necessary component of it.

There is a lot of confusion between Edge Computing and IOT. If stated simply, Edge Computing is nothing but the intelligent Internet of things(IOT) in a way. Edge Computing actually complements traditional IOT. In the traditional model of IOT, all the devices, like sensors, mobiles, laptops etc are connected to a central server. Now let's imagine a case where you give the command to your lamp to switch off, for such simple task, data needs to be transmitted to the cloud, analyzed there and then lamp will receive a command to switch off. Edge Computing brings computing closer to your home, that is either the fog layer present between lamp and cloud servers is smart enough to process the data or the lamp itself.

If we look at the below image, it is a standard IOT implementation where everything is centralized. While Edge Computing philosophy talks about decentralizing the architecture.

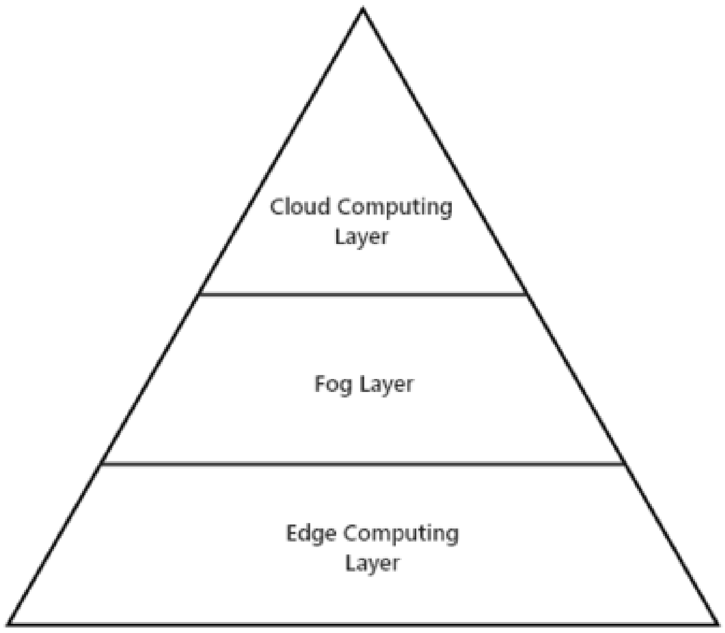


The Fog

Sandwiched between the edge layer and cloud layer, there is the Fog Layer. It bridges the connection between the other two layers.

The difference between fog and edge

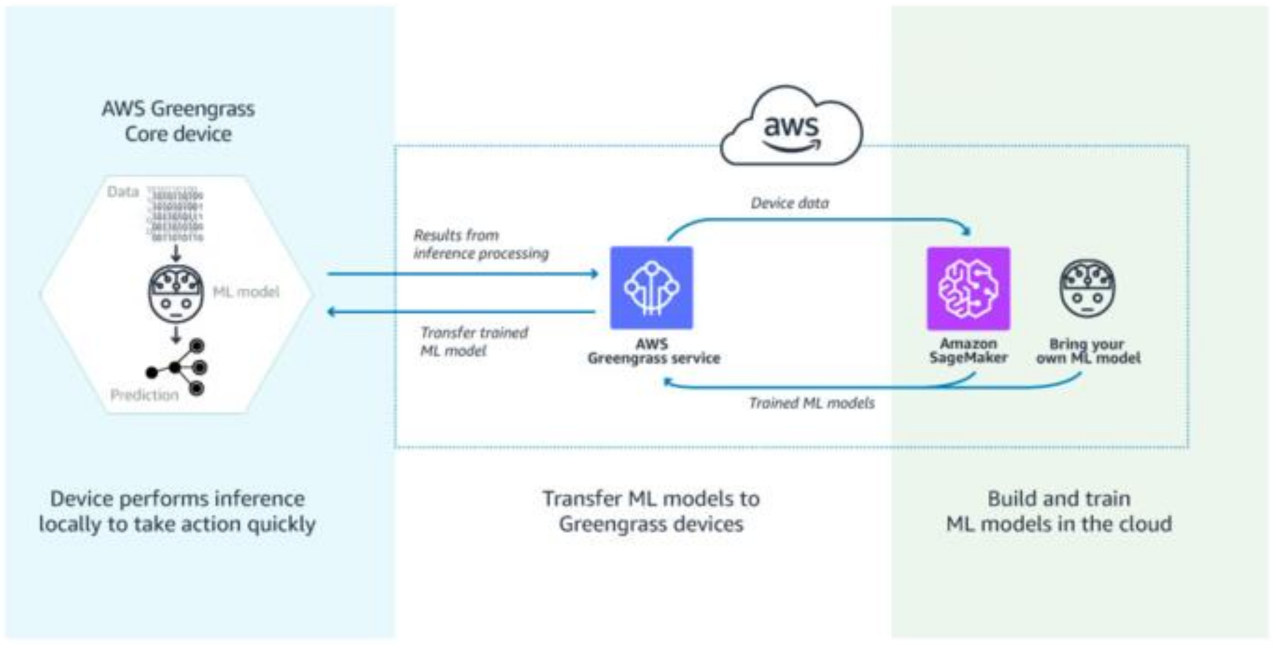
- Fog Computing — Fog computing pushes intelligence down to the local area network level of network architecture, processing data in a fog node or IoT gateway.
- Edge computing pushes the intelligence, processing power and communication capabilities of an edge gateway or appliance directly into devices like programmable automation controllers (PACs).



How do we manage Edge Computing?

The Device Relationship Management or DRM refers to managing, monitoring the interconnected components over the internet. AWS IOT Core and AWS Greengrass, Nebbiolo Technologies have developed Fog Node and Fog OS, Vapor IO has OpenDCRE using which one can control and monitor the data centers.

Following image (source — AWS) shows how to manage ML on Edge Computing using AWS infrastructure.


AWS Greengrass Core device
Data
ML model
Prediction
aws
Device data
Results from inference processing
Transfer trained ML model
AWS Greengrass service
Amazon SageMaker
Bring your own ML model
Trained ML models
Device performs inference locally to take action quickly
Transfer ML models to Greengrass devices
Build and train ML models in the cloud

AWS Greengrass makes it possible for users to use Lambda functions to build IoT devices and application logic. Specifically, AWS Greengrass provides cloud-based management of applications that can be deployed for local execution. Locally deployed Lambda functions are triggered by local events, messages from the cloud, or other sources.

This GitHub repo demonstrates a traffic light example using two Greengrass devices, a light controller, and a traffic light.

Conclusion

We believe that next-gen computing will be influenced a lot by Edge Computing and will continue to explore new use-cases that will be made possible by the Edge.

References

- https://www.forbes.com/sites/janakirammsv/2017/09/15/demystifying-edge-computing-device-edge-vs-cloud-edge/2/#5a547a605d19
- https://hackernoon.com/edge-computing-a-beginners-guide-8976b6886481
- https://www.automationworld.com/fog-computing-vs-edge-computing-whats-difference
- https://en.wikipedia.org/wiki/Edge_computing

- https://a16z.com/2016/12/16/the-end-of-cloud-computing/
- https://github.com/aws-samples/aws-greengrass-samples/tree/master/traffic-light-example-python

- “The key difference between the two architectures is exactly where that intelligence and computing power is placed,” he said. According to Newton:
- Fog computing pushes intelligence down to the local area network level of network architecture, processing data in a fog node or IoT gateway.
- Edge computing pushes the intelligence, processing power and communication capabilities of an edge gateway or appliance directly into devices like programmable automation controllers (PACs).
- “Many in industry indeed use the terms fog computing and edge computing (or edge processing) interchangeably,” said King. “Edge computing is actually an older expression that predates the fog computing term. By way of background, Cisco created the term fog computing years ago to describe a layer of computing at the edge ofthe network that could allow pre-processed data to be quickly and securely transported to the cloud. While Cisco certainly mastered the secure transport aspects of fog computing from the earliest days of IoT, very little has been done until recently to effectuate the data processing aspects of fog computing in real world IIoT use cases.”